ALGORITHMIE ET PROGRAMMATION

Correction de l'exercice LDR

Lien activité TinkerCAD

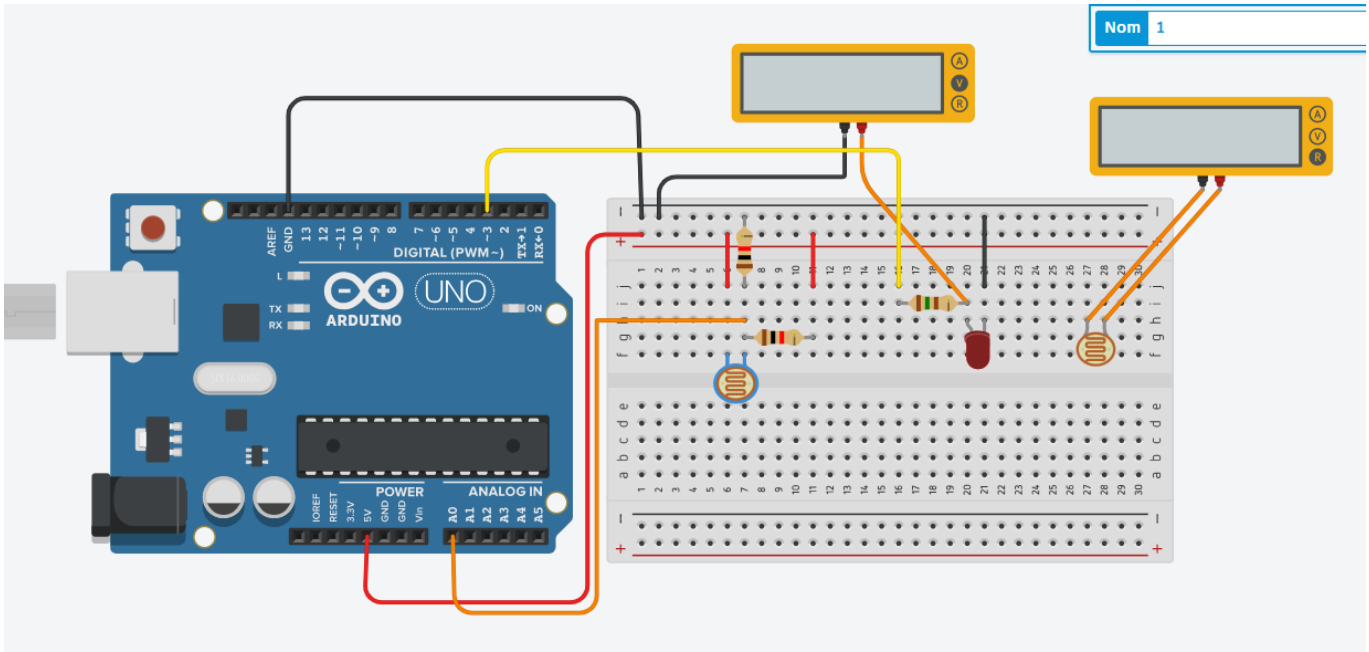
```
// C++ code
//
void setup()
{
  pinMode(3,OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  int ldr = analogRead(A0);
  float lux = 4.705 * ldr - 2648;

  if (lux < 200) {
    digitalWrite(3,HIGH);
  } else if (lux > 600) {
     digitalWrite(3,LOW);
  }

  Serial.print("Tension LDR: ");
  Serial.print(ldr);
  Serial.print(" , LUX: ");
  Serial.println(lux);

}
```



Arduino Vittascience.pdf

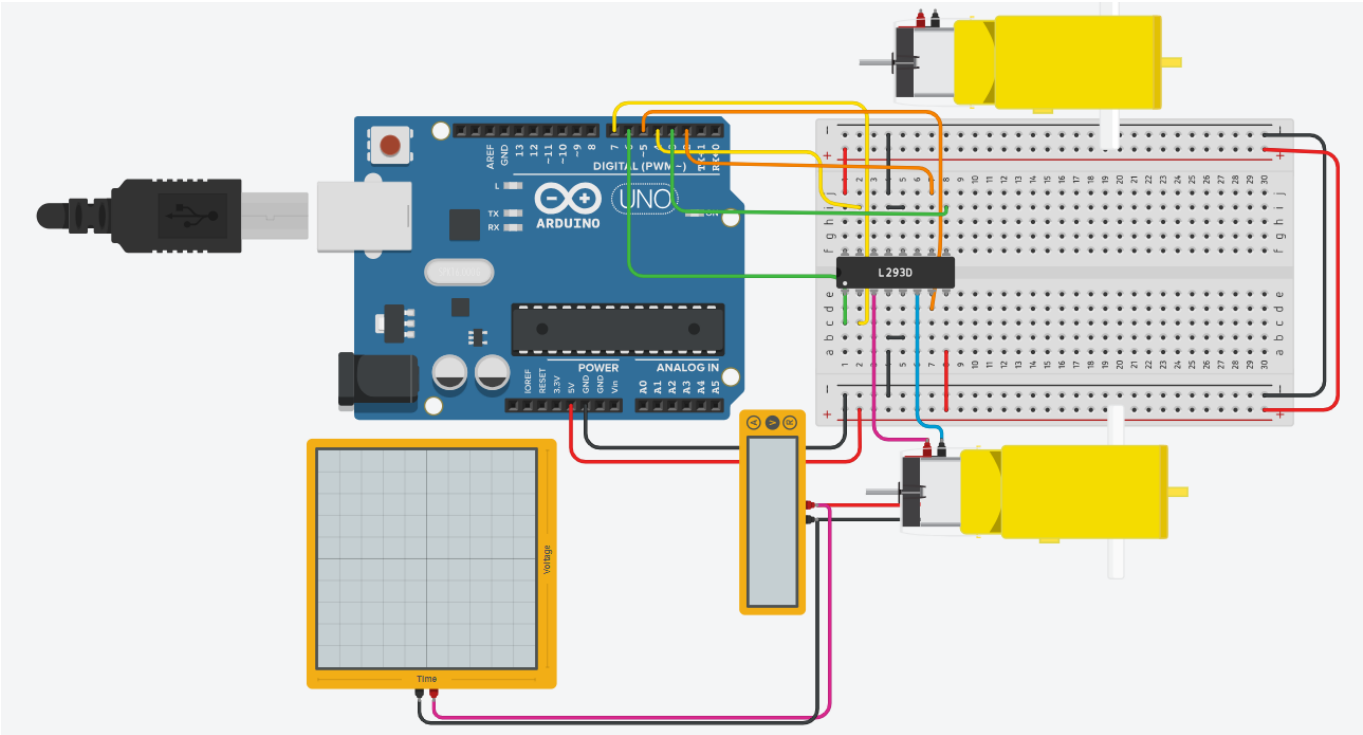
Site VittaScience

Site Wokwi

Circuits électroniques

Piloter un moteur avec arduino

https://www.tinkercad.com/joinclass/R9FK9NZYE



Objectifs attendus:

1. Comprendre comment la vitesse est contrôlée grâce à la technologie PWM ;
2. inverser le sens de rotation d'un moteur en jouant sur les valeurs MOTEUR1_IN1 et MOTEUR2_IN2;
3. faire un graphique Vitesse, Tension en faisant varier le paramètre transmis à MOTEUR1_EN de 0 à 255 (dans ce code il est à 62);
4. brancher le deuxième moteur et proposer le code pour tourner à gauche, à droite, avancer, reculer.

Se rendre sur la classe TinkerKad:

```
// C++ code
//
#define MOTEUR1_EN 6
#define MOTEUR1_IN1 7
#define MOTEUR1_IN2 5
```

```
void avancer() {
  analogWrite(MOTEUR1_EN, 62);
  digitalWrite(MOTEUR1_IN1, HIGH);
  digitalWrite(MOTEUR1_IN2, LOW);
  delay(200);

}

void setup() {
  pinMode(MOTEUR1_EN, OUTPUT);
  pinMode(MOTEUR1_IN1, OUTPUT);
  pinMode(MOTEUR1_IN2, OUTPUT);
}

void loop() {
  avancer();
  delay(500);
}
```

Fonctionnement du contrôleur moteur L293D:

https://www.tinkercad.com/things/8gpHB1obZhl-pir-sensor-example

TP ARDUINO:

tp_arduino_2023.pdf

initiation_arduino_2023.zip

tableau numération

familles_de_capteurs.pdf

Ressource pour la numération

numeration.pdf

https://pedagogie.ac-limoges.fr/sti_si/accueil/FichesConnaissances/Sequence2SSi/co/S2A25-Numeration-codage.html

Lien de la classe: https://www.tinkercad.com/joinclass/R9FK9NZYE

Prendre activité interrupteur crepusculaire et faire l'activité suivante:

```
Algorithme : Interrupteur crépusculaire avec détection de présence et
forçage
Variables (conceptuelles)
```

```
Luminosité : valeur mesurée par la LDR

Présence : état du capteur PIR (oui / non)

Forçage : état de l’interrupteur (activé / non)

État_LED : allumée / éteinte

Seuil_lumière : valeur limite entre jour et nuit

Algorithme principal
1. Initialisation

Fixer un seuil de luminosité (crépuscule).

Mettre la LED à l’état éteint.

2. Boucle de fonctionnement (répétée en permanence)

Mesurer la luminosité avec la LDR.

Lire l’état du capteur PIR (présence ou non).

Lire l’état de l’interrupteur de forçage.

3. Décision d’allumage

Cas 1 : Forçage activé

Allumer la LED, quelle que soit la luminosité et la présence.

Sinon (mode automatique) :

Si la luminosité est faible (nuit ou crépuscule)
ET qu’une présence est détectée
→ Allumer la LED.

Sinon
→ Éteindre la LED.

4. Retour au début de la boucle
```

Versions améliorées

Pour éviter que la LED s’éteigne trop vite :

- Si une présence est détectée, maintenir la LED allumée pendant un certain temps (temporisation). Puis revenir au mode normal.
- Ajouter un seuil haut et un seuil bas (hystérésis) pour la détection lumineuse pour que la lampe ne clignote pas de trop pas.

- Prévoir un mode économique qui permet de ne pas consommer trop d'électricité.